面向软件安全性需求分析过程的追踪模型

郑培真[1] 苑春春[1] 刘 超[1] 吴 际[1] 杨海燕[1] 胡 宁[2]
(北京航空航天大学计算机学院 北京 100191)[1] (中航工业西安航空计算技术研究所 西安 710065)[2]

摘 要 追踪性即关联一些制品及其中各种相关要素的机制或能力。安全关键系统开发不仅包括一般系统的开发过程,更重要的是必需要有独立的安全性分析,建立并验证系统的安全性需求。目前针对安全性分析过程的追踪性研究较少。安全相关标准如ARP-4761和DO 178C等提供了安全性分析过程的指导意见,然而其由于涉及的概念和方法很多,因此在实际应用和研究中常会忽略对一些关键信息的追踪。此外,软件安全性需求分析不仅应考虑系统到软件的安全性分析,还应考虑软件到系统的安全性分析。面向软件安全性需求分析过程建立安全性相关信息的双向追踪,有助于了解安全性需求的前因后果,为验证工作和影响分析提供便利。参照标准,构建面向软件安全性需求分析过程的追踪模型。

关键词 追踪性,软件安全性需求,安全性分析
中图法分类号 TP311 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.04.007

Traceability Model Oriented to Software Safety Requirement Analysis Process

ZHENG Pei-zhen[1] YUAN Chun-chun[1] LIU Chao[1] WU Ji[1] YANG Hai-yan[1] HU Ning[2]
(School of Computer Science and Engineering, Beihang University, Beijing 100191, China)[1]
(Xi'an Aeronautics Computing Technique Research Institute, AVIC, Xi'an 710065, China)[2]

Abstract Traceability is the mechanism or the ability to relate artefacts and the attached factors. Safety-critical system development, besides the general system development, contains more independent safety analysis which generates and verifies system safety requirements. At present, there are few traceability researches oriented to safety analysis process, which are of extremely challenging. Safety related standards, such as ARP-4761, DO 178C, provide guidelines for conducting safety analysis. However, some information may be neglected since there are a lot of concepts and methods. Besides, software safety requirement analysis should include both system to software and software to system safety analysis. Establishing bi-directional traceability of safety related information oriented to software safety requirement analysis process helps to simplify the verification and impact analysis. In this paper, we established a traceability model oriented to software safety requirement analysis process.

Keywords Traceability, Software safety requirements, Safety analysis

1 前言

追踪性是关联数据的一种方法。建立需求追踪有助于了解需求之间的依赖关系,便于变更管理和验证工作。特别是安全关键系统,其安全性需求混杂于功能需求和性能需求中,需要通过安全性分析过程独立识别,该过程的追踪性研究更具挑战性。

安全性相关标准,如ARP-4761[1],DO-178C[2],IEC 61508[3],DEF STAN 00-56[4]等,都指出安全性需求来自于安全性分析过程,其中标准ARP-4761, IEC 61508, DEF STAN 00-56等给出了安全性分析过程的指导意见,并要求建立追踪关系,但在实际应用和研究中,其常忽略标准中强调的人、假设、决策基本原理(rationales)等信息。安全关键系统常出现组件重用和系统演化发展。每次对系统的改变都要求对安全性重新进行评估,每次对组件的重用都要求对其环境重新进行评估。安全性分析过程产生的结果大部分是通过头脑风暴或专家经验确定的。如果这个过程中涉及的人、假设、原因、危害、需求之间没有建立起关联,在重新评估时就可能遗漏某些信息,造成错误评估,引起严重后果。

此外,Menon和Kelly[5]指出,系统到软件的安全性分析建立从系统危害到软件安全性需求的正向追踪。而在复杂系统的开发中,软件安全性分析的结果也会影响系统的开发,因此还应开展软件到系统的安全性分析,建立从软件安全性故障到系统危害的逆向追踪,而这恰恰是很多研究所忽略的。

到稿日期:2015-11-30 返修日期:2016-02-27 本文受某国家重点科研项目(MJ-S-2012-05)资助。
郑培真(1990-),女,硕士,主要研究领域为软件需求,E-mail:hf_cherish@163.com;苑春春(1985-),男,博士,主要研究领域为软件安全性等;刘 超(1958-),男,博士,教授,主要研究领域为软件测试等;吴 际(1974-),男,博士,副教授,主要研究领域为软件安全性与可靠性等;杨海燕(1974-),女,硕士,讲师,主要研究领域为软件测试等;胡 宁(1978-),男,博士,高级工程师,主要研究领域为机载安全关键软件等。

因此，本研究的目标是：依据标准，提取软件安全性需求分析过程的核心实体和关联，包括危害、安全性需求、决策基本原理、假设、人等，构建安全性追踪模型，以明确安全性需求的来源。这里，软件安全性需求分析过程包括从系统到软件的分析和从软件到系统的分析。

本文第 2 节评估相关研究；第 3 节识别本研究针对的用户、分析过程，介绍一般开发过程和软件安全性需求分析过程的步骤和关系；第 4 节识别软件安全性需求分析过程的重要实体及关联，构建追踪模型；第 5 节介绍实际应用案例的追踪模型；最后进行总结并介绍未来研究方向。

2　相关研究

Bashir 和 Qadir[6] 定义了 6 个追踪性评估准则，包括边界（系统或软件）、覆盖范围（起源与需求、需求与需求、需求与其他制品、其他制品间）、工具支持程度、技术是否得到验证、详细级别（是否考虑非功能性需求）、是否领域相关。他们对 1993 年到 2005 年提出的追踪性技术开展了调研，调研结果显示：大部分研究都是针对非领域相关的软件，并未涉及起源和需求的追踪链，且忽略了非功能需求。Knethen 等[7] 也指出：追踪性方法不仅应当捕获和管理文档实体之间的关系，还应当捕获设计过程中相关人员之间的协作，即利益相关者对开发的贡献，强调设计原理（design rationales）的记录。设计原理描述为什么选择某个特定的设计方案，为什么其他方案被否定，以及哪些需求被该决定所满足。Winkler 和 Pilgrim[7] 调研了需求工程领域的追踪性，将追踪性分为两类：功能性追踪（产品数据的转换）和非功能性追踪（包括理由、上下文、决策、技术），并总结了 8 类追踪链。

Ramesh 和 Jarke[8] 考虑高端用户和低端用户的追踪性需求。低端用户的任务是需求分解和分配，并支持其一致性验证和变更控制，因此他们对需求追踪的定义是需求到设计的文档实体转换。高端用户则要求覆盖所有的用户和客户，关心决策及基本原理，因此其对追踪性的定义是能够增加系统满足所有客户需求的能力。该研究关注的主要实体是需求、设计、决策基本原理、验证规程，将关联分为 satisfaction，evolution，rational，dependency 四大类，并采用引用模型建立追踪性模型。但是，该项研究针对的是一般系统的开发过程，并未考虑针对安全性分析过程的追踪性中存在的特殊性，如危害等实体。

Katta[9-11] 基于 Papadopoulos 和 McDermid[12] 提出的系统开发、评估、认证通用过程和 IEC 61508 标准，从安全性工程师和安全性案例工程师的角度出发，建立了开发过程和安全性分析过程的追踪概念模型，并依据概念模型提出了一种追踪方法 SaTrAp[13]，其目标是实现数据和关联概览以及影响分析。Katta 等[6] 提出的概念模型涵盖系统安全性分析的整个过程，包括软件安全性分析，但是未考虑软件安全性分析到系统安全性分析的追踪，以及决策基本原理和假设的追踪。此外，其提出的 SaTrAp 方法目前也仅考虑了系统安全性分析部分。Lee 和 Howard 等人[14] 提出的 SpecTRM 关注系统需求和规格说明，强调设计原因、假设的记录及追踪关系的建立。该研究虽然是安全性驱动的，但是对安全性分析过程仅提出了一系列分析方法，不强调这个过程应当追踪的实体和关联。Peraldi-Frati 和 Albinet[15] 关注安全关键系统中安全性需求的获取和验证，建立安全性需求和其他各种产品之间的追踪链。然而此研究对安全性分析过程的建模粒度不够，仅仅捕捉危害和安全性需求两个数据，忽略了假设、决策基本原理、失效等数据。

综上所述，目前的追踪模型都不能满足安全关键软件需求工程师的追踪性需求。构建面向软件安全性需求分析过程的追踪模型，不仅考虑系统到软件的安全性分析，还考虑软件到系统的安全性分析，以及决策基本原理和假设等实体，是亟待解决的一个重要课题。此模型可为软件安全性需求建模和验证的后续研究提供基础。

3　目标和安全性分析过程

Knethen 等[16] 指出建立追踪模型之前，首先应确定建立追踪性的目标和要追踪的过程。本研究关注安全关键软件需求工程师的追踪性需求。安全关键软件需求工程师的一个任务是开发软件高层安全性需求，要求实现从系统安全性分析输出到软件安全性分析输出的追踪。另外，需要对软件执行失效进行分析，并将新派生的安全性需求和功能需求提交给系统安全分析师评估，要求实现从软件安全性分析输出到系统安全性分析输出的逆向追踪。

IEC 61508 详细规定了整个安全生命周期的活动，包括：1）概念开发期安全性活动；2）系统安全性生命周期；3）软件安全性生命周期；4）整体验证等主要活动。DO-178C 指出，软件安全性相关需求是从系统安全性评估过程中获得的。ARP-4761 详细定义了系统安全性评估的整个过程，主要用于安全性需求的生成和验证，通过评估系统功能和设计来确定系统危害是否都已被解决。安全性评估过程主要分为 3 个阶段：1）概念和体系结构开发期，执行飞机级功能危害分析和故障树分析；2）初步设计期，执行系统级功能危害分析和故障树分析；3）详细设计期，执行项目级失效模式影响分析和故障树分析等。

结合上述 3 个标准，捕捉安全性分析和系统开发的 3 个重要阶段，其过程可抽象为图 1。本研究重点关注安全性需求的生成。安全性分析过程是开发过程的内在过程，两者关联紧密。系统概念开发阶段（D1）通过对系统功能执行初期的功能危害分析（S1）获得系统安全性目标、系统危害、危害分类的原因等主要数据。在系统设计阶段（D2），系统架构逐渐清晰，可以执行系统安全性分析（S2），以初期识别的危害为树顶，构建故障树，分析造成系统危害的失效和故障，并提出相应的安全性需求处理失效。设计的变化也可能派生新的安全性需求。同时，需要对细化的子系统功能需求进行重新评估，分析其可能产生的失效，并向上追踪到系统危害。这一阶段识别安全性需求，并将安全性需求分配到软硬件，同时验证提出的系统架构是否满足安全性需求和初期提出的安全性目标。软件开发阶段（D3），一方面，开发实现从系统安全性分析过程分配的安全性需求；另一方面，执行软件安全性分析（S3），采用类似系统安全分析的过程对软件执行故障树分析和软件失效模式影响分析。图中括号内标识的内容表示可采

用的现有方法，加粗线条标识的是自下而上追踪链，虚线将过程和输出分离开。

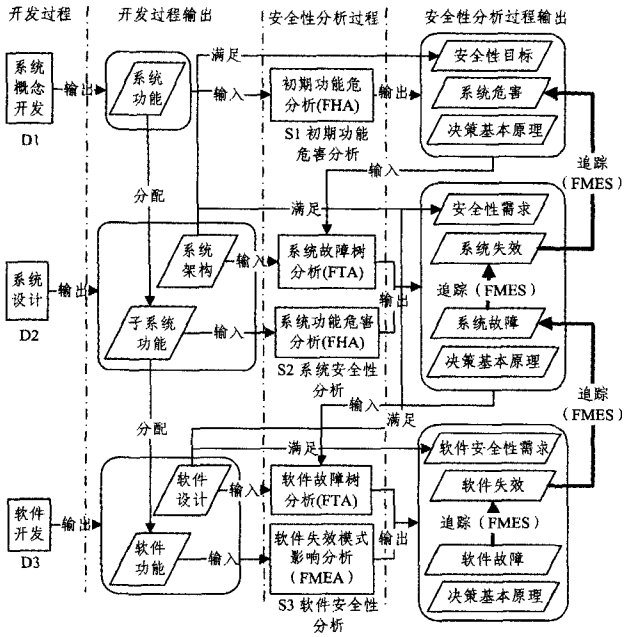

图 1 开发过程和安全性过程

4 追踪模型

4.1 实体

根据图 1，参考标准 ARP-4761，识别安全性分析 3 个阶段的输入输出实体，如表 1 所列。

表 1 安全性分析过程的相关实体

安全性过程	输入	输出
初期功能危害分析(preliminary functional hazard analysis)	1. 系统高层功能需求(HighLevelFunctionReqs) 2. 高层目标(HighLevelGoals，即高层非功能需求) 3. 最初系统架构决定(initialSystemArchitecture)	1. 危害(hazards) 2. 损害(harm，即危害的影响) 3. 危害严重程度(severity) 4. 危害发生概率(probability) 5. 风险(risk) 6. 安全性目标(safetyGoals) 7. 假设(assumptions) 8. 决策基本原理(rationales)
系统安全性分析(system safety analysis)	1. 系统危害(hazards) 2. 安全性目标(safetyGoals) 3. 细化的系统功能需求(FunctionReqs) 4. 细化的系统架构(systemArchitecture)	1. 系统失效(systemFailures) 2. 系统组件故障(systemFaults) 3. 安全性需求(SafetyReqs) 4. 设计决策(Design Decisions) 5. 假设(assumptions) 6. 决策基本原理(rationales)
软件安全性分析(software safety analysis)	1. 软件高层安全性需求(SwhighLevelSafetyReqs) 2. 软件功能需求(SWFunctions)	1. 软件安全性需求(SWSafetyReqs) 2. 软件功能失效(SWFailures) 3. 软件功能组件故障(SWfaults) 4. 假设(assumptions) 5. 决策基本原理(rationales)

对于上述实体，有以下几点说明。

(1)概念

1)安全性(Safety)：风险可接受的状态[ARP-4754：2010]。

2)风险(risk)：损害发生的概率以及损害的严重程度[ISO/IEC Guide 51：1999，definition 3.2]，[IEC 61508-4：2010]，[ARP-4761：1996]。

3)损害(harm)：人身伤害，财产损失或对环境造成的危害[ISO/IEC Guide 51：1999，definition 3.3]，[IEC 61508-4：2010]。

4)危害(hazard)：①造成 harm 的潜在根源[ISO/IEC Guide 51：1999，definition 3.5]，[IEC 61508-4：2010]；②由失效、故障、外部事件、错误或者这些组合造成的一种潜在的不安全状况[APR-4761：1996]。

5)失效(failure)：功能缺失，或者系统或系统某部分发生故障[ARP-4761.1996]。

6)故障(fault)：项目或系统的一个非预期异常。故障的发生可能导致失效[SAE ARP-4761，1996]，[DO-178C]。

其中，危害、失效、故障的关系可进一步解释为：危害强调系统的一种直接表现，这种表现是一种不安全状况，如飞机指令显示屏显示数据错误；失效强调功能的不正确执行或缺失，如返回指令的功能返回错误数据；故障强调功能某个组件错误的运行表现，如功能运行时出现数组指针越界访问。实际分析过程中，初期分析提取的许多危害并不是真正的系统层危害，只是危害的原因(失效或故障)。正确地依据概念识别的各类数据，一方面能提供清晰的缺陷数据关联，另一方面有助于收集和分析各抽象层次的经验数据。

(2)强调人为原因造成的危害，忽略其他缺陷

将造成损害的所有原因统称为缺陷，可分为 3 类：1)错误，强调人为因素，包括硬件错误和软件错误；2)随机硬件失效，强调硬件的不可抗的老化、损坏等；3)外部失效，如火灾等。硬件失效防护和外部环境保护通常不受控，因此软件相关安全性分析通常只考虑第一项，即人为造成的缺陷，例如人为编码错误、需求开发错误、设计错误等。

4.2 关联

追踪模型规定应当追踪的实体以及关联。根据安全关键软件需求工程师的目标，结合 4.1 节提取的实体构建安全性追踪模型，如图 2 和图 3 所示，其术语与表 1 括号内的英文术语一致。

(1)建立自上而下的追踪

建立自上而下的追踪主要是关注系统到软件的安全性分析，始于系统概念开发期。每个高层功能需求与零个或多个危害存在关联关系(FHA，HAZOP 分析)。每个危害存在影响(即损害)、发生概率、严重程度等属性。危害的这些属性决定了它的风险，而风险威胁系统的安全性。为了控制风险，保证安全性，提出安全性需求来处理危害。系统开发初期提出的安全性需求主要确立了系统安全性目标，如完整性或可靠性的一些需求和设计约束。而每个安全性功能需求必然与某个安全性目标相关联。

基于初期功能危害分析确定的危害和安全性目标做出设计决策，以最终满足系统的安全性需求。系统设计阶段，从危害出发，识别造成危害的系统失效和故障。为处理这些失效和故障，安全性需求被提出。至此，自上而下的追踪关系建立完成。

(2)建立自下而上的追踪

自下而上建立追踪主要是关注软件到系统的安全性分析，目的是建立从新识别的软件失效到系统危害的逆向追踪，即如果该软件失效是原来已识别的系统故障，建立逆向追踪

即可；否则，将软件失效作为新的系统故障，评估其影响，即系统失效，向上继续评估系统失效影响，追踪到危害。至此，自下而上的追踪关系建立完成。

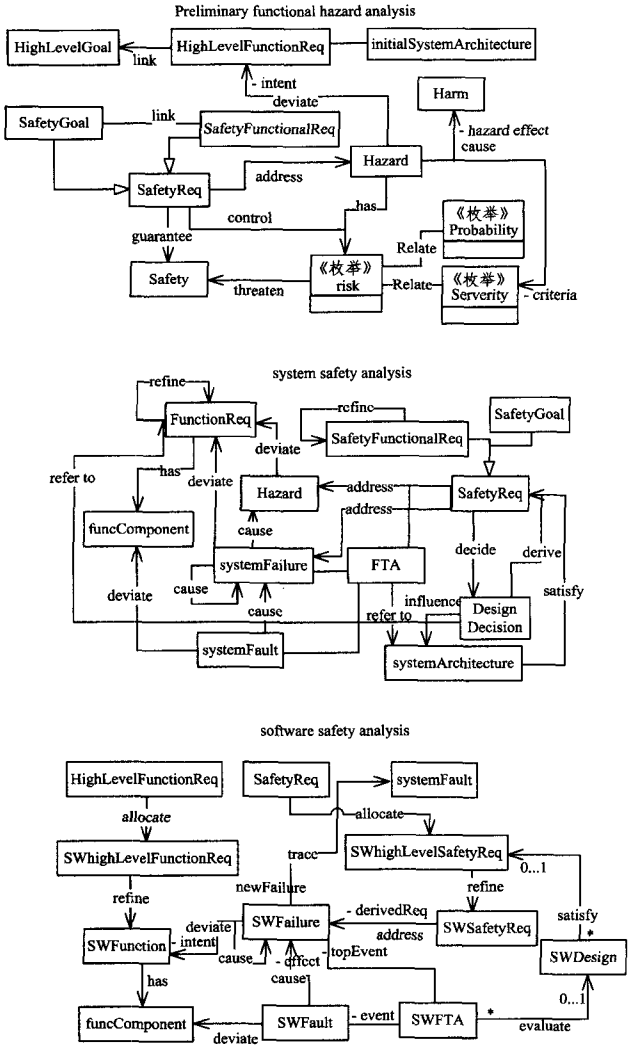

图 2　安全性实体追踪模型

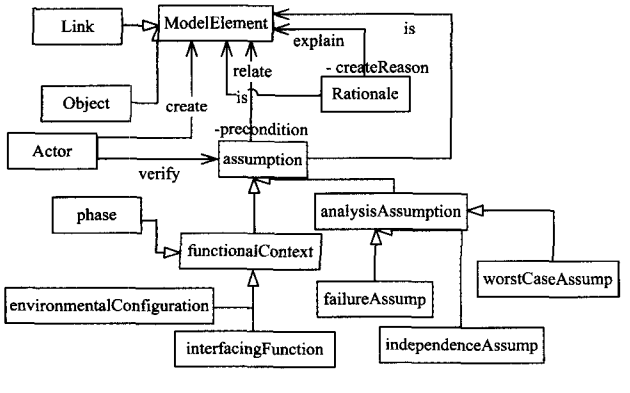

图 3　角色、假设、决策基本原理的追踪模型

(3)建立假设、决策基本原理、角色的追踪

如图 3 所示，将图 2 和图 3 中的所有实体定义为 Object 类的子类，关联定义为 Link 类的子类。那么，1)在分析的任一步骤，对任一数据或关联，都可能会做出假设，即其前提条件。假设应当被验证。假设可分为功能上下文假设(functionalContext)和安全性分析假设(analysisAssumption)。功能上下文假设包括运行阶段(phase)、环境配置和状态(environmentalConfiguration)等，解释功能运行环境；安全性分析假设包括失效假设、最坏情况假设和独立性假设。例如，某两个潜在失效的发生会导致一个系统失效，那么在对该系统失效执行故障树定量分析时，常常会做这样的失效假设：这两个组件在任务之初均未失效。再例如，某软件的数据在两个显示器上显示，这两个显示器显示的数据是相同的，那么对于产生错误数据的软件失效，可能做如下最坏情况假设：该软件失效同时影响两个显示器。2)在分析的任一步骤，任一数据或关联的创建都有其原因(rationale)，特别是一些设计决策。3)任一数据或关联均由某个人创建。用户角色(actor)活动的记录和追踪有利于任务的分配和管理。

(4)追踪约束关系

建立双向追踪链时，应遵循的约束包括：

1)每个危害都被处理。危害的处理可能是直接的，即该危害直接被某个或某些安全性需求所处理；也可能是间接的，即导致该危害的所有失效或故障都被某个或某些安全性需求所处理，最终使得该危害被处理。

2)每个失效/故障，或者被处理，或者由于某种理由放弃处理。失效/故障的处理可能是直接的，即该失效/故障直接被某个或某些安全性需求所处理；也可能是间接的，即导致该失效/故障的所有失效或故障都被某个或某些安全性需求所处理，最终使得该失效被处理。

3)每个被间接处理的危害/失效/故障，都要求提供证据(assumption 类)来证明导致该危害/失效/故障的所有失效或故障已识别完备。

4)每个放弃处理的失效/故障都要求指明理由(rationale 类)。

5)每个安全性目标(safetyGoal)都关联到某个或某些安全性功能需求(safetyFunctionalReq)。

6)每个假设(assumption)都最终被某个或某些人(actors)确认。

7)对于每个危害、失效和故障，都应分析其影响、发生概率、严重程度。

5　应用案例

本节以用于航电系统的某机载操作系统为例进行说明。该操作系统支持 FC 通信系统、综合显示系统、导航系统、航向姿态系统等的软件运行。其中 FC 通信任务由 FC 通信子卡驱动软件与 FC 接口硬件协同完成。FC 通信子卡驱动软件运行于操作系统之上，主要提供设备管理、通信管理、时统管理、网络管理、中断管理和配置加载的功能，完成 FC 网络数据的收发，为应用系统提供 FC 网络通信能力。FC 通信系统的软件架构图如图 4 所示。

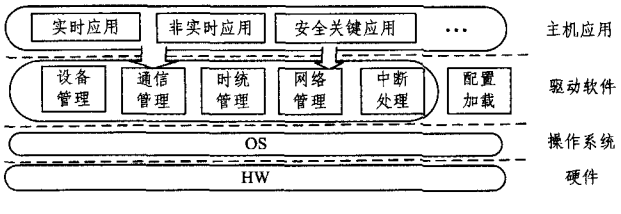

图 4　FC 通信系统软件架构图

以通信管理功能为例，通信管理实现消息收发的功能，支持 128 条数据块消息和 16 条流消息的传输控制。对该功能执行危害分析，并对底层操作系统执行安全性分析，从而得到与消息接收功能相关的安全性追踪模型，如图 5 所示。

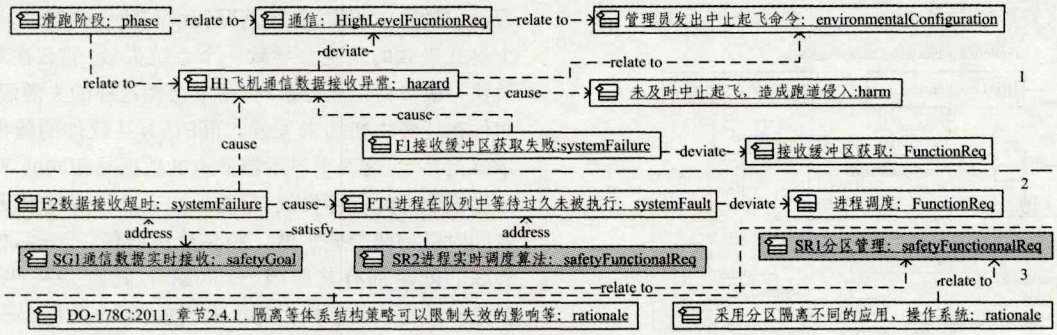

图 5 通信管理安全性相关信息追踪模型

第 1 部分构建自上而下的追踪关系，对应初步功能危害分析和系统安全性分析，建立了危害、失效、功能、假设的追踪关联。初期识别到的危害 H1“飞机通信数据接收异常”是“通信”功能的一种失效情况。由于通信子卡驱动软件的通信管理功能与数据接收有关，因此该软件的失效 F1“接收缓冲区获取失败”可能导致上述危害 H1。

第 2 部分构建自下而上的追踪关系，对应软件安全性分析。“进程调度”功能在运行时可能出现故障 FT1“进程在队列中等待过久未被执行”，该故障可能导致系统失效 F2“数据接收超时”，并进一步导致危害 H1。失效 F2 是软件安全性分析过程新识别到的系统失效，也是导致危险 H1 的原因。

第 3 部分构建安全性需求和决策基本原理的关联，对应决策基本原理等数据的追踪（见图 3）。这是利用历史经验数据直接获取操作系统高层安全性需求的特例，着重捕获提出该需求的决策基本原理。

结束语 追踪性在安全性关键系统开发中发挥着重要的作用。目前针对安全性分析过程的追踪性研究较少，且常忽略软件失效到系统危害的逆向追踪；此外，标准强调的决策基本原理、假设、人等数据也常被忽略。因此，本文研究从安全关键软件需求工程师的角度出发，依据 ARP-4761，DO 178C，IEC 61508 标准，描述系统和软件安全性分析过程，构建安全性追踪模型，分析其中的重要实体和关联。依据此模型，将进一步研究软件安全性需求建模和验证。

参 考 文 献

[1] Guidelines and methods for conducting the safety assessment process on civil airborne systems and equipment: ARP 4761[S]. Society of Automotive Engineers, December 1996.

[2] Software Considerations in Airborne Systems and Equipment Certification: DO 178C[S]. 2011.

[3] Functional safety of electrical/electronic/programmable electronic safety-related systems: IEC 61508[S]. International Electrotechnical Commission, April 2010.

[4] Safety management requirements for defence systems containing programmable electronics: Defence standard 00-56[S]. Ministry of Defence, UK, 1995.

[5] MENON C, KELLY T. Eliciting software safety requirements in complex systems[C]// 2010 4th Annual IEEE Systems Conference. IEEE, 2010: 616-621.

[6] BASHIR M F, QADIR M A. Traceability Techniques: A Critical Study[C]// Multitopic Conference, 2006 (INMIC'06). IEEE, 2006: 23-24.

[7] WINKILER S, PILGRIM J V. A survey of traceability in requirements engineering and model-driven development[J]. Software and Systems Modeling, Springer Berlin, 2010, 9(4): 529-565.

[8] RAMSESH B, JARKE M. Towards reference models for requirements traceability[J]. IEEE Transactions on Software Engineering, IEEE Press, Piscataway, 2001, 27(1): 58-93.

[9] KATTA V, STÅLHANE T. A conceptual model of traceability for safety systems[C]// Eletronic Proc. 2nd Complex Systems Design & Management Conference (CSD&M 2011). Paris, France, 2011.

[10] PASPOTNING C, KARPATI P, KATTA V. Enhancing CHASSIS: A Method for Combining Safety and Security[C]// Unpublished, International Workshop on Security in Air Traffic Management and other Critical Infrastructures (SecATM2013). 2013.

[11] RASPOTING C, KARPATI P, KATTA V. CHASSIS Guideline (draft)[OL]. (2013-06-01). https://bora.uib.no/handle/1956/6172.

[12] PAPADOPOULOS Y, MCDERMID J A. The potential canfor a generic approach to certification of safety critical systems in the transportation sector[J]. Reliability Engineering and Systems Safety, Elsevier Science, 1999, 63(1): 47-66.

[13] KATTA V, STÅLHANE T. Traceability of safety systems: approach, meta-model and tool support: tech. report HWR-1053 [R]. OECD Halden Reactor Project, Institute for Energy Technology, 2013.

[14] LEE G, HOWARD J, ANDERSON P. Safety-critical requirements specification and analysis using spectrm[C]// Proceedings of the 2nd Meeting of the US Software System Safety Working Group. Feb 2002.

[15] PERALDI-FRATI M A, ALBINET A. Requirement traceability in safety critical systems[C]// EDCC2010 - Workshop on Critical Automotive Applications: Robustness and Safety (CARS' 2010). 2010: 11-14.

[16] KNETHEN A V, PAECH B. A survey on tracing approaches in practice and research: Research Report, ESE-Report, 095.01/E [R]. Fraunhofer IESE, Kaiserslautern, 2002